Amendments to the Claims:

This listing of claims will replace all prior versions and listings of claims in this application:

Listing of Claims:

1. (Currently amended) A processor implemented method of identifying ~~dividing data into predominantly fixed-size chunks so that~~ duplicate data chunks in a data stream by dividing the data stream into fixed-size chunks, the method ~~are identified~~, comprising:

sliding a window of a fixed size, over the data chunks;

tracking a residue data that has been slid over by the window and that has not yet been emitted as a chunk; and

testing if a data within the window has been previously seen; and

upon determination that the data within the window has been previously seen, emitting as one or more unique chunks the residue data that has been slid over, and further emitting the data in the window as a duplicate chunk.

2. (Canceled)

3. (Currently amended) The method of claim [[2]] 1, wherein testing if the data within the window has been previously seen comprises testing whether the data within the window is likely to have been seen before.

4. (Currently amended) The method of claim [[2]] 1, further comprising sliding the window forward by a predetermined number of bytes.

5. (Currently amended) The method of claim [[2]] 1, further comprising sliding the window by a number of bytes that is determined by an offset from a beginning of the chunk, the offset being associated with a marker in the data within the window.

6. (Original) The method of claim 5, wherein the marker comprises a predetermined pattern that appears substantially consistently throughout the data.

7. (Original) The method of claim 1, further comprising emitting as one or more chunks, residue data that has been slid over by the window but not yet emitted, if the size of the residue data exceeds the size of the block.

8. (Original) The method of claim 1, further comprising remembering the chunk by computing a plurality of functions relative to the data in the chunk, and further remembering resulting computed values.

9. (Original) The method of claim 8, further comprising testing whether the data in the window is likely to have been seen before by computing one or more of the plurality of functions relative to the data; and

further checking to determine if the computed value is one of the resulting computed values that have been remembered.

10. (Original) The method of claim 9, wherein testing whether the data in the window is likely to have been seen before comprises performing a first least accurate and least expensive test; and

then performing a second more accurate and more expensive test only if the first test is positive.

11. (Original) The method of claim 9, wherein remembering the chunk further comprises looking for predetermined patterns in the chunk, and remembering the occurrences of the predetermined patterns and corresponding offsets from the beginning of the chunk.

12. (Original) The method of claim 1, wherein testing whether the data in the window is likely to have been seen before further comprises looking for the patterns in the data.

13. (Original) The method of claim 12, further comprising checking to determine if the occurrences of any the predetermined patterns in the data in the window have been remembered.

14. (Original) The method of claim 13, further comprising looking up the corresponding offset that has been remembered, when the patterns have been remembered.

15. (Original) The method of claim 14, further comprising sliding the window by using the remembered offsets to line up the window with a previously seen chunk.

16. (Original) The method of claim 11, wherein looking for specific patterns in the data is performed by computing a mathematical

function over subsequences of the data and finding computed values with certain values.

17. (Original) The method of claim 16, wherein remembering the occurrences of the specific patterns is implemented by computing a supplemental mathematical function of data near the specific patterns, and by remembering corresponding computed values.

18. (Currently amended) The method of claim [[2]] 1, wherein testing if the data within the window has been previously seen comprises determining whether the data within the window has been previously seen within a preceding period of time.

19. (Currently amended) The method of claim [[2]] 1, wherein testing if the data within the window has been previously seen comprises determining whether the data within the window has been previously seen within a predetermined amount of preceding data.

20. (Original) The method of claim 3, wherein testing if the data within the window has been previously seen comprises determining whether the data within the window has probably been previously seen within any of a preceding period of time or within an amount of preceding data.

21. (Currently amended) A processor implemented system of identifying ~~dividing data into predominantly fixed-size chunks so that~~ duplicate data in a data stream by dividing the data stream into fixed-size chunks, the system ~~are identified~~, comprising:

a window of a fixed size, that is slid over the data chunks;

a residue data that has been slid over by the window, that has not yet been emitted as a chunk, and that is tracked; and

a test module that determines if a data within the window has been previously seen; and

upon determination that the data within the window has been previously seen, the residue data that has been slid over is emitted as one or more unique chunks, and the data in the window is emitted as a duplicate chunk.

23. (Currently amended) The system of claim [[22]] 21, wherein the test module tests whether the data within the window is likely to have been seen before.

24. (Currently amended) The system of claim [[22]] 21, wherein the window is slid forward by a predetermined number of bytes.

25. (Currently amended) The system of claim [[22]] 21, wherein the window is slid by a number of bytes that is determined by an offset from a beginning of the chunk, the offset being associated with a marker in the data within the window.

26. (Original) The system of claim 25, wherein the marker comprises a predetermined pattern that appears substantially consistently throughout the data.

27. (Original) The system of claim 21, wherein the residue data that has been slid over by the window but not yet emitted, is emitted as one or more chunks if the size of the residue data exceeds the size of the block.

28. (Original) The system of claim 21, wherein the chunk is remembered by computing a plurality of functions relative to the data in the chunk, and wherein resulting computed values are also remembered.

29. (Original) The system of claim 28, wherein the testing module tests whether the data in the window is likely to have been seen before by computing one or more of the plurality of functions relative to the data, and further checks to determine if the computed value is one of the resulting computed values that have been remembered.

30. (Original) The system of claim 29, wherein the test module performs a first least accurate and least expensive test, and then performs a second more accurate and more expensive test only if the first test is positive.

31. (Currently amended) The system of claim [[22]] 21, wherein the test module determines whether the data within the window has been previously seen within any of a preceding period of time or within an amount of preceding data.

32. (Currently amended) A computer program product having executable instruction codes stored on a computer readable medium, for identifying ~~dividing data into predominantly fixed-size chunks so that~~ duplicate data chunks in a data stream by dividing the data stream into fixed-size chunks, the computer program product ~~are identified~~, comprising:

a first set of instruction codes for sliding a window of a fixed size, over the data chunks;

a second set of instruction codes for tracking a residue data that has been slid over by the window and that has not yet been emitted as a chunk; and

a third set of instruction codes for testing if a data within the window has been previously seen; and

upon determination that the data within the window has been previously seen, a fourth set of instruction codes emits as one or more unique chunks, the residue data that has been slid over, and further emits as a duplicate chunk the data in the window.

34. (Currently amended) The computer program product of claim [[33]] 32, wherein the third set of instruction codes tests whether the data within the window is likely to have been seen before.

35. (Currently amended) The computer program product of claim [[33]] 32, further comprising a fifth set of instruction codes for sliding the window forward by a predetermined number of bytes.

36. (Currently amended) The computer program product of claim [[33]] 32, wherein the first set of instruction codes slides the window by a

number of bytes that is determined by an offset from a beginning of the chunk, the offset being associated with a marker in the data within the window.

37. (Original) The computer program product of claim 36, wherein the marker comprises a predetermined pattern that appears substantially consistently throughout the data.

38. (Original) The computer program product of claim 32, wherein the fourth set of instruction codes emits as one or more chunks, residue data that has been slid over by the window but not yet emitted, if the size of the residue data exceeds the size of the block.

39. (Original) The computer program product of claim 32, further comprising a storage for remembering the chunk by computing a plurality of functions relative to the data in the chunk, and remembering resulting computed values.

40. (Original) The computer program product of claim 39, wherein the third set of instruction codes tests whether the data in the window is likely to have been seen before by computing one or more of the plurality of functions relative to the data; and

further determines if the computed value is one of the resulting computed values that have been remembered.

41. (Original) The computer program product of claim 40, wherein the third set of instruction codes tests whether the data in the window is likely

to have been seen before by performing a first least accurate and least expensive test; and

then performs a second more accurate and more expensive test only if the first test is positive.

42. (Original) The computer program product of claim 32, wherein the third set of instruction codes tests whether the data within the window has been previously seen within any of a preceding period of time or within an amount of preceding data.